RRAML: Reinforced Retrieval Augmented Machine Learning

Andrea Bacciu[1], Florin Cuconasu[1], Federico Siciliano[1], Fabrizio Silvestri[1], Nicola Tonellotto[2] and Giovanni Trappolini[1,*]

[1]*Sapienza University of Rome*
[2]*University of Pisa*

Abstract

The emergence of large language models (LLMs) has revolutionized machine learning and related fields, showcasing remarkable abilities in comprehending, generating, and manipulating human language. However, their conventional usage through API-based text prompt submissions imposes certain limitations in terms of context constraints and external source availability. LLMs suffer from the problem of hallucinating text, and in the last year, several approaches have been devised to overcome this issue: adding an external Knowledge Base or an external memory consisting of embeddings stored and retrieved by vector databases. In all the current approaches, though, the main issues are: (i) they need to access an embedding model and then adapt it to the task they have to solve; (ii) in case they have to optimize the embedding model, they need to have access to the parameters of the LLM, which in many cases are "black boxes". To address these challenges, we propose a novel framework called Reinforced Retrieval Augmented Machine Learning (RRAML). RRAML integrates the reasoning capabilities of LLMs with supporting information retrieved by a purpose-built retriever from a vast user-provided database. By leveraging recent advancements in reinforcement learning, our method effectively addresses several critical challenges. Firstly, it circumvents the need for accessing LLM gradients. Secondly, our method alleviates the burden of retraining LLMs for specific tasks, as it is often impractical or impossible due to restricted access to the model and the computational intensity involved. Additionally, we seamlessly link the retriever's task with the reasoner, mitigating hallucinations and reducing irrelevant and potentially damaging retrieved documents. We believe that the research agenda outlined in this paper has the potential to profoundly impact the field of AI, democratizing access to and utilization of LLMs for a wide range of entities.

Keywords

Deep Learning, Information Retrieval, Large Language Models

1. Introduction

The advent of Large Language Models (LLMs) has brought about a paradigm shift in machine learning and its related disciplines. LLMs [1, 2, 3, 4, 5] have exhibited unprecedented capabilities in understanding, generating, and manipulating the human language. Famously, ChatGPT [3] has entered the public space by reaching one million users in a matter of days. The way these models are used is through API that only allows submitting a textual prompt and getting back from the server the generated text. However, this causes an immediate limitation: all

22nd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2023) - Discussion Papers
[*] Corresponding author.
trappolini@diag.uniroma1.it (G. Trappolini)

information must be passed through this context, and we know transformer-based models do not scale nicely. Even if they did, API costs are charged on the basis of their usage. Therefore, using long contexts would be expensive. Even if one had the resources to run their own LLM, the costs of training and of the hardware infrastructure, and the environmental impact should be considered. There is an impendent need, though, to accommodate the enormous power of those models to specific user needs by making sure that they could use the reasoning capabilities of LLMs, through in-context learning [1] on their data.
A solution is to adopt a retrieval-augmented approach [6, 7]. In this setting, a retriever is used to filter out relevant information to be passed as context to the reasoner. This generates a new problem, however, namely that the retriever and the reasoner are not aligned [8, 9, 10]. In particular, the retriever might not be trained on the task of interest to the user. Moreover, the retriever might actually provide "dangerous" pieces of information to the reasoner, as proved in [11], leading to poor results and, more importantly, to hallucinations.

Ideally, one would have to fine-tune these models to account for these issues. Within this setting, fine-tuning the model for a given task is technically impossible. We asked ourselves: "*Is it still possible to use the API that gatekeeps those powerful LLMs on our data without the need for fine-tuning?*" We show that this question has a positive answer and in this paper, we propose a novel framework, Reinforced Retrieval Augmented Machine Learning (RRAML), in which we combine the reasoning capabilities of large foundational models enhanced by the provision of supporting relevant information provided by a retriever that searches them in a large database. In this setting, an efficient retriever model is tasked to search for relevant information in an arbitrarily large database of data provided by users. Once this set of relevant data has been retrieved, it is forwarded to the reasoner (a large foundational model such as ChatGPT, for instance) through its API to "reason" on the input and produce an adequate result. In particular, we plan to overcome current limitations, namely that the retriever's task is detached from that of the reasoner, reducing in such a way the tendency of LLM to hallucinate and diminishing the number of damaging documents (as defined in [12, 13, 8]) returned by the retriever. The approach we devise in this research work exploits recent advances in reinforcement learning. Recently, in fact, reinforcement learning techniques like PPO [14] have been used to improve large foundational models with human feedback where the loss is non-differentiable. We propose to link the training phase of the retriever to the final task outcome by the use of a purposefully crafted reward model that depends either on human feedback or on the specific characteristics of the task data. The RL technique also offers the advantage of not requiring fine-tuning an LLM as a reasoner, which can be considered a black box in this setting, and exchanged freely.

Finally, we argue that the research agenda we lay out in this paper has the potential to hugely impact the field of AI and democratize the access and use of these large foundational models to a large set of entities.

2. Methodology

The system takes as input a task description, a query, and a database and gives as output the response generated by a reasoner. The overall system architecture, shown in Figure 1, consists of three main components: a Generative Language Model, a Retriever, and a Reasoner (typically an LLM).

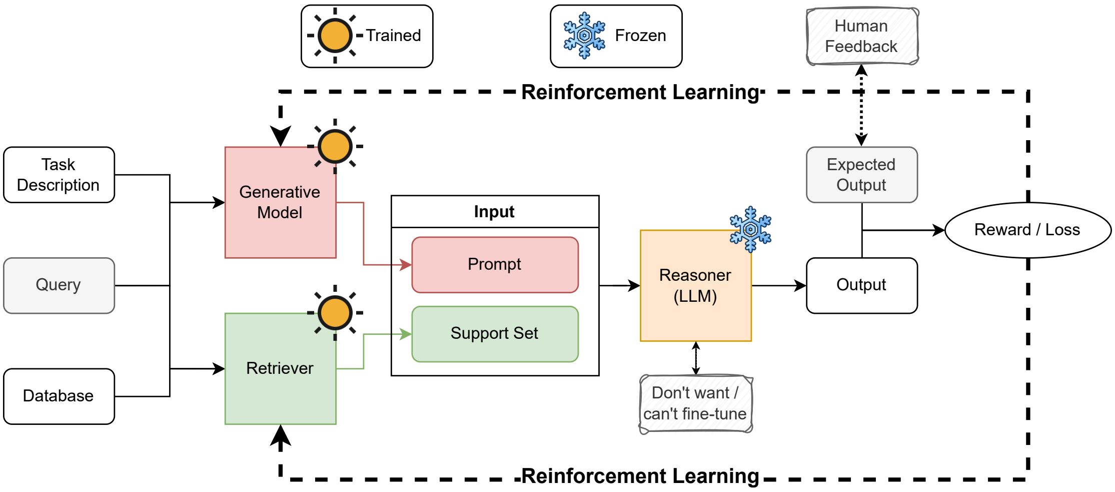

Figure 1: High-level design of the RRAML framework. On the left side, there are the three inputs: Task Description, user's query, and a database that represents the external knowledge used to augment/update the reasoner. Then, we present the overall architecture flow with the Retriever, Generative Language Model, and Reasoner. Finally, how the reward is computed and propagated in the Generative Language Model and Retriever.

More in detail, the Generative Language Model takes the *task description* and *query* as input and generates a prompt. The Retriever takes the *query* and the *database* as input and outputs a support set, which is then concatenated with the *query* and passed to the Reasoner.

2.1. Data

The data is a critical component of the framework: the *task description* guides the generation of an appropriate prompt, the *query* represents the user request, and the *database* provides the data needed by the reasoner to perform the task.

Task Description The *task description* is a string that defines the nature of the task, possibly with expected results, that the user wants to perform. For example, if the user wants to generate a summarization of multiple news articles, a possible *task description* could be "News Summarization". If the user wants to perform question answering on a vast document collection, the *task description* could be "Question Answering".

Query The *query* represents the user's need. The Retriever will operate on the *database* w.r.t to the user's query, and the resulting data is input for the task. For example, if the user wants to summarize a collection of news articles, the *query* could be the topic the user is interested in. If the user wants to answer a specific *query*, this becomes the actual question.

Database The *database* is a collection of public or private data (or documents) that can be queried to provide relevant information to satisfy the user's information needs. The database represents the knowledge needed by the Reasoner to perform the task. The data stored in the *database* will depend on the specific task and may include text, images, audio, and other data types (as in [8]). For example, if the user wants to summarize multiple news articles, the *database* could be an indexed collection of articles. If the user wants to perform Question Answering, the *database* may consist of facts related to a particular topic (as in [9, 10]).

2.2. Models

Generative Language Model The Generative Language Model component of the framework is responsible for generating textual instructions based on the input *Task Description* and *Query* that maximize the rewards w.r.t Reasoner. Specifically, it receives a string representing the task to be performed (*Task Description*) and a query (*Query*) that represents the user's request. The Generative Language Model then generates a textual prompt that is relevant to the query and the task by performing automatic prompt engineering.

Retriever The Retriever component of the framework is responsible for retrieving relevant data from the Database based on the user's query. We refer to the Retriever outputs as support set (as in [9, 10]). A support set is a subset of the data from the Database that either directly answers the given query or contributes to the final answer.

Prompt Aggregator This component is responsible for processing the input required by the *Reasoner.* In its simplest form, it just needs to concatenate the prompt generated by the Generative Language Model with the Support Set provided by the Retriever. However, in a more complex version, it may need to rework the prompt based on the number of support sets received to ensure that the LLM can provide a coherent response. For example, if the Retriever provides two support sets, the Prompt Aggregator may need to split the prompt into two parts and concatenate each part with one of the support sets.

Reasoner The Reasoner is responsible for generating the answer to the user's query based on the final prompt generated by the Prompt Aggregator. The Reasoner can be a pre-trained model like GPT or a custom-trained model specific to the task at hand. The output of the LLM is a textual response, which can be further parsed to comply with the intended output.

2.3. Reinforcement Learning

The Reinforcement Learning (RL) part of the framework is responsible for fine-tuning the Generative Language Model (GLM) and Retriever based on the computed reward. The RL is a crucial part of RRAML, it will be used to constantly improve the GLM and Retriever. As mentioned earlier, the retriever will get a penalty if some of his recommendations will leads the Reasoner to a hallucinate, for example by adding damaging documents. The RL allows use to integrate and augment the signals in the training of these models, going beyond the data present in their training set, ensuring that they are aligned with the environment (i.e., the reasoner and the final task).

Reward The reward function can be defined based on the similarity between the generated output and the expected output and it can be estimated by training a Reward Model [14].

RL algorithm The specific RL method which can be used is Deep Q-Networks (DQN) [15], which is a model-free RL algorithm that learns to maximize the cumulative reward over time. DQN combines Q-Learning, which is a RL algorithm that learns the optimal action-value function, with a Deep Neural Network to approximate the action-value function. In the proposed framework, DQN is used to train the Generative Language Model and the Retriever to maximize the reward obtained from user feedback. The update process is performed by backpropagating the reward signal through the neural networks using Stochastic Gradient Descent (SGD). The weights of the neural networks are updated in the direction that maximizes the expected reward, using the Q-Learning update rule. The update is performed iteratively until convergence, which is achieved when the expected reward stops improving.

Human-in-the-loop Human preferences can be incorporated into our ML system by allowing users to provide feedback on the system's output. This feedback will be used to compute the reward for the RL algorithm and will help improve the performance of the overall system over time. We acknowledge that some tasks may not have a clear expected output or may require additional context that is not available in the input data. In these cases, we will leverage human-in-the-loop approaches to provide additional context and guidance to the system. For example, crowd-sourcing platforms or internal subject matter experts can be used to provide feedback on the system's output and help train the model on more complex tasks.

3. Use Case Example

RRAML promises to be effective in many applications. Consider a situation where a company possesses a private database, which consists of factual information expressed in natural language, and they need to apply reasoning to this data. The volume of their data may exceed the context capacity of the LLM, and fine-tuning is not an option, for pricing/environmental impact or because the LLM is served by other company APIs. To tackle this challenge, RRAML uses its retriever to get only the relevant facts within the context, enabling the LLM to reason over them.

For instance, suppose a company has an employee list, projects that employees are currently or were previously assigned to, and performance evaluation grids with text-based feedback from superiors. The company might want to assign employees to a new project on a specific topic. To do so, it is necessary to input the information contained in these data to the LLM. However, due to capacity constraints, the entire data cannot fit within the context. Therefore, the retriever has to return a subset of this information, perhaps excluding data on projects from the distant past, employees who are already overburdened with multiple projects, or employees who have never worked on a project related to the same topic.

4. Related Work

Recent years have seen the emergence of large language models. Starting from the first Generative Pre/Training Model, better known as GPT [16], these kinds of large language models have rapidly improved. Even further, deep learning models have now reached multimodal capabilities beyond just images, with methods proficient on audio [17, 18, 19], video [20, 21], and 3D [22, 23, 24]. GPT-4 [25] is the most recent iteration, but in the meanwhile, many have rushed to propose their own version. Google has recently released BARD[1], while Meta has proposed their own take on LLM with LLaMA [4]. The research community has also capitalized its effort by releasing several open source LLM of different sizes, like Bloom [2], Dolly[2], and RWKV [26]. However, all these models fail to scale to a larger context size, either by excessive computational costs or by “losing it in the middle”, as shown in [27].

To address this context-length limitation, some have tried to incorporate external knowledge into LLMs [28, 29, 30]. In particular, in “Retrieval-enhanced machine learning” [31], authors have envisioned a framework in which retrieval systems can enhance the performance of a machine learning model. More recently, there have been attempts of jointly training retrieval models with LLMs [6, 32], notably, the line of research on neural databases, in which the authors tried to replace a traditional database with a neural framework removing the need for a schema [10, 9, 8]. However, all these works assume full access to the reasoner module, which is not the case for most users in practice.

To overcome this limitation, many have tried to craft systems that are able to deliver an optimized prompt that is input to the LLM. For instance, the research conducted by [33] demonstrated a substantial influence of the sequence in which prompts are presented on the ultimate performance of the task. Meanwhile, a study by Nie et al. [34] highlighted that the performance is susceptible to the arrangement of the examples in the prompt, prompt templates, and the in-context instances in the prompt. Lester et al. [35] suggested a method to enhance task performance by adding adjustable tokens during fine-tuning. LLM-AUGMENTER iteratively revises [30] to improve the model response.

All the works introduced above do not improve on the retriever, which is assumed fixed. In our work, we propose to finetune the retriever in conjunction with the reasoner to improve on results. Since the feedback is non-differentiable we resort to reinforcement learning. In particular, recent formulation such as Proximal Policy Optimization (PPO) [36] make use of a differentiable neural reward module to include and account for generally non-differentiable feedback, like in the case of reinforcement learning with human feedback (RLHF).

5. Conclusions

In conclusion, RRAML provides a promising framework for building intelligent interfaces to interact with large language models like GPT. By combining a generative language model with a retriever, this approach can effectively improve the performance of language models and help them understand user intents better.

[1] https://bard.google.com/

[2] https://github.com/databrickslabs/dolly

However, this approach also comes with several challenges and uncertainties, such as the need for a large amount of training data, the potential for bias in the data and models, and the difficulty of balancing the trade-offs between generative and retrieval-based approaches.

Despite these challenges, RRAML holds great promise for creating more intelligent, natural, and effective interfaces for interacting with language models. We hope that this paper has provided a useful overview of this approach and its potential applications, and we look forward to further research and development in this exciting area.

Acknowledgments

This work was supported by PNRR MUR projects PE0000013-FAIR, SERICS (PE00000014), IR0000013-SoBigData.it.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[2] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, et al., Bloom: A 176b-parameter open-access multilingual language model, arXiv preprint arXiv:2211.05100 (2022).

[3] OpenAI, Chatgpt: A large-scale language model for conversational ai, OpenAI Blog (2022).

[4] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample, Llama: Open and efficient foundation language models, 2023. arXiv:2302.13971.

[5] A. Bacciu, G. Trappolini, A. Santilli, E. Rodolà, F. Silvestri, Fauno: The italian large language model that will leave you senza parole!, arXiv preprint arXiv:2306.14457 (2023).

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.

[7] Z. Xie, S. Singh, J. McAuley, B. P. Majumder, Factual and informative review generation for explainable recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, 2023, pp. 13816–13824.

[8] G. Trappolini, A. Santilli, E. Rodolà, A. Halevy, F. Silvestri, Multimodal neural databases, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 2619–2628. URL: https://doi.org/10.1145/3539618.3591930. doi:10.1145/3539618.3591930.

[9] J. Thorne, M. Yazdani, M. Saeidi, F. Silvestri, S. Riedel, A. Halevy, Database reasoning over text, arXiv preprint arXiv:2106.01074 (2021).

[10] J. Thorne, M. Yazdani, M. Saeidi, F. Silvestri, S. Riedel, A. Halevy, From natural language processing to neural databases, in: Proceedings of the VLDB Endowment, volume 14, VLDB Endowment, 2021, pp. 1033–1039.

[11] A. Sauchuk, J. Thorne, A. Y. Halevy, N. Tonellotto, F. Silvestri, On the role of relevance in natural language processing tasks, in: E. Amigó, P. Castells, J. Gonzalo, B. Carterette, J. S. Culpepper, G. Kazai (Eds.), SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022, ACM, 2022, pp. 1785–1789. URL: https://doi.org/10.1145/3477495.3532034. doi:10.1145/3477495.3532034.

[12] D. Carmel, N. Cohen, A. Ingber, E. Kravi, Ir evaluation and learning in the presence of forbidden documents, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 556–566.

[13] A. Sauchuk, J. Thorne, A. Halevy, N. Tonellotto, F. Silvestri, On the role of relevance in natural language processing tasks, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1785–1789.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347 (2017).

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, nature 518 (2015) 529–533.

[16] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., Improving language understanding by generative pre-training, OpenAI Blog (2018).

[17] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, I. Sutskever, Jukebox: A generative model for music, arXiv preprint arXiv:2005.00341 (2020).

[18] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, D. Grangier, M. Tagliasacchi, et al., Audiolm: a language modeling approach to audio generation, IEEE/ACM Transactions on Audio, Speech, and Language Processing (2023).

[19] G. Barnabò, G. Trappolini, L. Lastilla, C. Campagnano, A. Fan, F. Petroni, F. Silvestri, Cycledrums: automatic drum arrangement for bass lines using cyclegan, Discover Artificial Intelligence 3 (2023) 4.

[20] Z. Luo, D. Chen, Y. Zhang, Y. Huang, L. Wang, Y. Shen, D. Zhao, J. Zhou, T. Tan, Videofusion: Decomposed diffusion models for high-quality video generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[21] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, C. Feichtenhofer, Mvitv2: Improved multiscale vision transformers for classification and detection, 2022. arXiv:2112.01526.

[22] Z. Chen, G. Wang, Z. Liu, Scenedreamer: Unbounded 3d scene generation from 2d image collections, 2023. arXiv:2302.01330.

[23] G. Trappolini, L. Cosmo, L. Moschella, R. Marin, S. Melzi, E. Rodolà, Shape registration in the time of transformers, Advances in Neural Information Processing Systems 34 (2021) 5731–5744.

[24] O. Halimi, I. Imanuel, O. Litany, G. Trappolini, E. Rodolà, L. Guibas, R. Kimmel, Towards precise completion of deformable shapes, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16, Springer, 2020, pp. 359–377.

[25] OpenAI, Gpt-4 technical report, 2023. arXiv:2303.08774.

[26] B. PENG, RWKV-LM, 2021. URL: https://github.com/BlinkDL/RWKV-LM. doi:10.5281/

zenodo.5196577.

[27] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, arXiv preprint arXiv:2307.03172 (2023).

[28] M. Ghazvininejad, C. Brockett, M.-W. Chang, B. Dolan, J. Gao, W.-t. Yih, M. Galley, A knowledge-grounded neural conversation model, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

[29] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, J. Weston, Wizard of wikipedia: Knowledge-powered conversational agents, arXiv preprint arXiv:1811.01241 (2018).

[30] B. Peng, M. Galley, P. He, H. Cheng, Y. Xie, Y. Hu, Q. Huang, L. Liden, Z. Yu, W. Chen, et al., Check your facts and try again: Improving large language models with external knowledge and automated feedback, arXiv preprint arXiv:2302.12813 (2023).

[31] H. Zamani, F. Diaz, M. Dehghani, D. Metzler, M. Bendersky, Retrieval-enhanced machine learning, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 2875–2886.

[32] Y. Zhang, S. Sun, X. Gao, Y. Fang, C. Brockett, M. Galley, J. Gao, B. Dolan, Retgen: A joint framework for retrieval and grounded text generation modeling, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 11739–11747.

[33] Y. Lu, M. Bartolo, A. Moore, S. Riedel, P. Stenetorp, Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity, arXiv preprint arXiv:2104.08786 (2021).

[34] F. Nie, M. Chen, Z. Zhang, X. Cheng, Improving few-shot performance of language models via nearest neighbor calibration, arXiv preprint arXiv:2212.02216 (2022).

[35] B. Lester, R. Al-Rfou, N. Constant, The power of scale for parameter-efficient prompt tuning, arXiv preprint arXiv:2104.08691 (2021).

[36] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph, A. Madry, Implementation matters in deep policy gradients: A case study on ppo and trpo, arXiv preprint arXiv:2005.12729 (2020).